Glycosylator: a Python framework for the rapid modeling of glycans

Lemmin, Thomas ; Soto, Cinque

Abstract: BACKGROUND Carbohydrates are a class of large and diverse biomolecules, ranging from a simple monosaccharide to large multi-branching glycan structures. The covalent linkage of a carbohydrate to the nitrogen atom of an asparagine, a process referred to as N-linked glycosylation, plays an important role in the physiology of many living organisms. Most software for glycan modeling on a personal desktop computer requires knowledge of molecular dynamics to interface with specialized programs such as CHARMM or AMBER. There are a number of popular web-based tools that are available for modeling glycans (e.g., GLYCAM-WEB (http:// https://dev.glycam.org/gp/) or Glycosciences.db (http://www.glycosciences.de/)). However, these web-based tools are generally limited to a few canonical glycan conformations and do not allow the user to incorporate glycan modeling into their protein structure modeling workflow. RESULTS Here, we present Glycosylator, a Python framework for the identification, modeling and modification of glycans in protein structure that can be used directly in a Python script through its application programming interface (API) or through its graphical user interface (GUI). The GUI provides a straightforward two-dimensional (2D) rendering of a glycoprotein that allows for a quick visual inspection of the glycosylation state of all the sequons on a protein structure. Modeled glycans can be further refined by a genetic algorithm for removing clashes and sampling alternative conformations. Glycosylator can also identify specific three-dimensional (3D) glycans on a protein structure using a library of predefined templates. CONCLUSIONS Glycosylator was used to generate models of glycosylated protein without steric clashes. Since the molecular topology is based on the CHARMM force field, new complex sugar moieties can be generated without modifying the internals of the code. Glycosylator provides more functionality for analyzing and modeling glycans than any other available software or webserver at present. Glycosylator will be a valuable tool for the glycoinformatics and biomolecular modeling communities.

DOI: https://doi.org/10.1186/s12859-019-3097-6

Lemmin, Thomas; Soto, Cinque (2019). Glycosylator: a Python framework for the rapid modeling of glycans. BMC Bioinformatics, 20(1):513.
DOI: https://doi.org/10.1186/s12859-019-3097-6

BMC Bioinformatics

SOFTWARE Open Access

Glycosylator: a Python framework for the rapid modeling of glycans

Thomas Lemmin[1,2*] and Cinque Soto[3,4*]

Abstract

Background: Carbohydrates are a class of large and diverse biomolecules, ranging from a simple monosaccharide to large multi-branching glycan structures. The covalent linkage of a carbohydrate to the nitrogen atom of an asparagine, a process referred to as *N*-linked glycosylation, plays an important role in the physiology of many living organisms. Most software for glycan modeling on a personal desktop computer requires knowledge of molecular dynamics to interface with specialized programs such as CHARMM or AMBER. There are a number of popular web-based tools that are available for modeling glycans (e.g., GLYCAM-WEB (http://https://dev.glycam.org/gp/) or Glycosciences.db (http://www.glycosciences.de/)). However, these web-based tools are generally limited to a few canonical glycan conformations and do not allow the user to incorporate glycan modeling into their protein structure modeling workflow.

Results: Here, we present Glycosylator, a Python framework for the identification, modeling and modification of glycans in protein structure that can be used directly in a Python script through its application programming interface (API) or through its graphical user interface (GUI). The GUI provides a straightforward two-dimensional (2D) rendering of a glycoprotein that allows for a quick visual inspection of the glycosylation state of all the sequons on a protein structure. Modeled glycans can be further refined by a genetic algorithm for removing clashes and sampling alternative conformations. Glycosylator can also identify specific three-dimensional (3D) glycans on a protein structure using a library of predefined templates.

Conclusions: Glycosylator was used to generate models of glycosylated protein without steric clashes. Since the molecular topology is based on the CHARMM force field, new complex sugar moieties can be generated without modifying the internals of the code. Glycosylator provides more functionality for analyzing and modeling glycans than any other available software or webserver at present. Glycosylator will be a valuable tool for the glycoinformatics and biomolecular modeling communities.

Keywords: *N*-linked glycosylation, Glycan modeling, Glycoprotein, Biomolecular modeling

Background

Glycosylation is an important post-translational modification of proteins, where a carbohydrate is covalently attached by an enzyme to specific amino acids motifs known as sequons space [1–4]. Glycosylation has several principal structural and functional roles in biology, that include protein folding [5], tissue repair [6], and cell migration [7]. In eukaryotes, nearly 70% of the proteome is believed to be glycosylated [8]. More recently, glycosylation has been observed in bacteria where it has been associated with their virulence and the formation of biofilms [9]. For viruses, such as HIV and Influenza, glycosylation allows for evasion of the host's immune system [10, 11]. Thus, determining the role of glycan structure in biology is essential in order to understand pathogenesis. The diverse and dynamic nature of glycan structures makes it difficult to resolve their structure experimentally through traditional approaches (e.g., x-ray crystallography, cryogenic electron microscopy (cryo-EM) or nuclear magnetic resonance (NMR)). Computational methods, such as molecular dynamics (MD) can help resolve glycan dynamics but this method is computationally intensive and cannot be used for the rapid modeling of glycan structure. Complementary techniques that are more rapid and available through a graphical user interface

* Correspondence: thomas.lemmin@inf.ethz.ch; cinque.soto@vumc.org
[1]DS3Lab, System Group, Department of Computer Sciences, ETH Zurich, CH-8093 Zurich, Switzerland
[3]Vanderbilt Vaccine Center, Vanderbilt University Medical Center, Nashville, TN 37232, USA
Full list of author information is available at the end of the article

(GUI) should allow users to gain new insights into glycan-protein structure.

In silico modeling of glycoprotein is a tedious and time consuming process and tools, such as CarbBuilder [12], POLYS [13], doGlycans [14], SWEET-II [15], GLYCAM-Web [16], Glycan Reader [17, 18] and CHARMM-GUI glycan modeler [19] were developed to facilitate the modeling of glycans. CarbBuilder, POLYS and doGlycans are open source programs that allow building glycan structures from their primary sequence of monosaccharide units. SWEET-II is part of the website Glycosciences.DB [20] and can be used to build 3D structures of glycans. Furthermore, the website provides a number of tools for manipulating and analyzing glycans. GLYCAM-Web offers several options that simplify the building and set-up of molecular dynamics simulations of glycoproteins. It uses the GLYCAM force field [21] that is compatible with the AMBER force field. Finally, Glycan Reader recognizes most types of glycans and their chemical modifications found in the Protein Data Bank (PDB), which are all available in the CHARMM force field [22]. It also provides the option for editing their three-dimensional structure. Glycan Modeler generates complex glycans and glycoconjugates by searching templates from a fragment database. Glycan Reader and Modeler have both been integrated into CHARMM-GUI [23], a powerful website widely employed for setting up molecular dynamics simulation. In addition, CHARMM-GUI provides the functionality for modelling glycolipids and lipopolysaccharides (LPS) and to combine them with complex biological membrane simulations [24]. While many of these tools are available as webservers making them ideal for their ease of use and distribution, this limits their ability to be customized for the specific needs of some users; for example, for tasks that require batch modeling of several glycoforms for a given protein or adding non-canonical saccharides to a protein structure.

We describe here Glycosylator, a Python framework designed for the rapid modeling of glycoprotein. It can be used directly in a Python terminal or script to identify, manipulate, and build glycans. In addition, the GUI allows for the quick visualization and modification of glycosylated proteins (such as one downloaded directly from the PDB). The molecular description of glycans is based on the CHARMM force field [22]. New saccharides appearing in updated versions of the force field or defined by the user can easily be added. Modeled glycans can be further refined by removing clashes and sampling alternate conformations. Since Glycosylator is distributed as a Python package, users can easily adapt the code to meet their specific needs.

Implementation

The Glycosylator framework is composed of 7 classes, several of which can be employed as standalone instances for other applications in molecular modeling (Additional file 1: Figure S1 in the Supporting Information (SI) section). At the core of Glycosylator is the Molecule class. A Molecule is defined as a single covalently linked set of atoms and is implemented around the ProDy [25] and NetworkX [26] packages. ProDy is widely used for studying biomolecules and offers several functions for storing and manipulating structures. The functions and classes provided are used in the Molecule class for saving and quickly accessing the structural data of a molecule. The topological properties of a molecule are represented here as a graph using the NetworkX package. A Molecule can be instantiated directly with a 3D structure (PDB) or using a MoleculeBuilder instance and the topology information provided for the CHARMM force field [22]. When loading a glycoprotein, Glycosylator will identify all *O*- and *N*-linked sequons and their glycans. The structure and topology of each of the glycans can then be modified. Clashes and alternative conformations for glycans can be optimized with the Sampler class. Finally, the graphical representation of glycans provided by the Drawer class makes use of Matplotlib [27], a Python package used for plotting. Taken together, Glycosylator provides more functionality for analyzing and modeling glycans than many popular software packages and webservers (Table 1). The main functions used for glycosylating a protein can be conveniently accessed through Glycosylator's GUI (Additional file 1: Figure S2).

Below, we briefly describe each class. Detailed examples for the usage of each class are provided in the Supporting Information (Additional file 1: Example S1) section and on the Github repository.

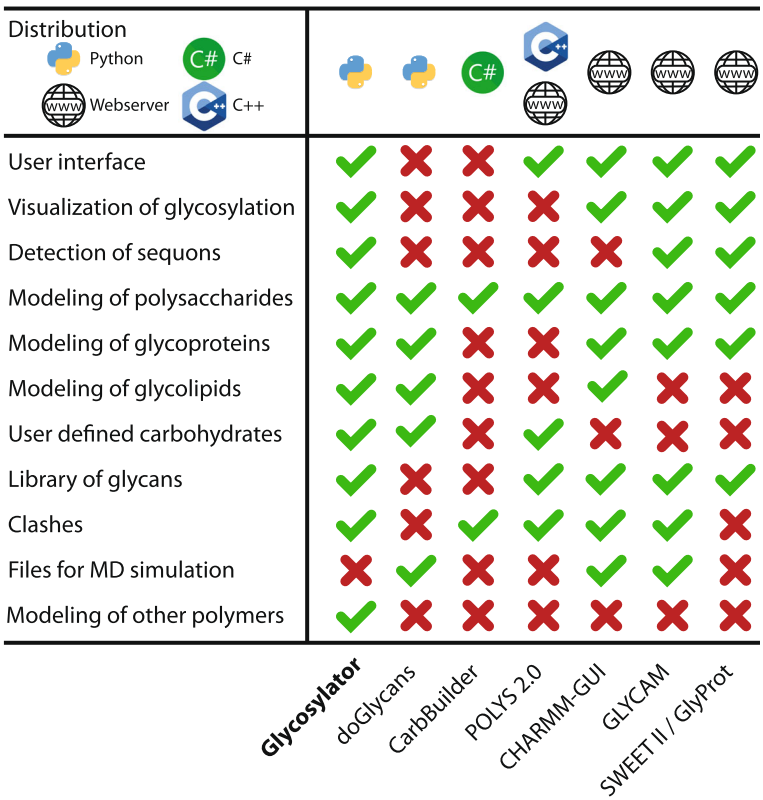
CHARMM classes

CHARMM force field topology and parameter files are parsed using the CHARMMTopology and CHARMMParameters classes, respectively. The data is stored in a dictionary for a quick and easy access. The CHARMMTopology class creates and stores an additional dictionary for looking up patches. The patches are used to define the glycosidic bonds between saccharide units and are required for modification (e.g., deleting atoms).

Molecule class

The Molecule class is used for storing the coordinates (Prody's AtomGroup) and connectivity (NetworkX graph) of a molecule. The bonds, angles and dihedrals are assigned either by the user or automatically based on the distances between atoms. The molecule's connectivity is saved as a directed graph. The user can provide the root atom to define the direction of the connectivity graph; by default, the first atom of the molecule is

Table 1 List of functionalities offered by the available software and webservers for modeling glycans. CHARMM-GUI includes Glycan Reader and Modelers, as well as the glycolipid and LPS modelers



chosen. Ring structures are automatically detected identifying all rotatable torsional angles that are not part of a cycle. These torsional angles can be measured, set to a specific value or rotated by a given amount. An inter-residue graph is also built in order to quickly parse through a molecule composed of several residues.

MoleculeBuilder class

The MoleculeBuilder class is employed for building and editing molecules. Information about the connectivity and atoms of a molecule are extracted from a CHARMMTopology instance. This class allows for the initialization of a Prody residue (AtomGroup). Applying a patch (CHARMM) will modify one or several residues. For glycans, patches are typically used to define the glycosidic linkage. MoleculeBuilder interfaces directly with the Prody AtomGroup and returns all the information required for creating a Molecule instance.

Glycosylator class

Glycosylator class was designed to deal specifically with glycans/glycoprotein. It can import a PDB file and automatically extract all the *O*- and *N*-linked sequons and associated atoms. Each glycan is saved as a Molecule instance in a dictionary. The key of the dictionary is the residue number and chain of the sequon. Glycosylator uses an internal text representation for storing a topology tree for each glycan structure. These trees describe the connectivity and saccharide units that compose a polysaccharide. A library of these structures can be imported into a Glycosylator instance or saved as a simple text file or a SQL database. Glycosylator can then compare the extracted connectivity tree to the internal dataset of known glycans to identify them based on the glycosidic linkage and residue type. We do note that chemical post-modifications of glycans are not supported in the current version. Glycans can be extended, trimmed or modeled ab initio. This can be achieved by providing the identification of a known oligosaccharide (in the library) or with a topology tree describing the connectivity and glycan units of the desired oligosaccharide. The topology tree is a string representation of a glycan.

Sampler class

Sampler class implements a genetic algorithm for removing clashes between Molecules and their environment (e.g., protein). The CHARMM force field energy function for the torsional angles will be used for biasing

the random number generator and to sample more energetically favorable torsional angles [22]. The generation of the initial population can be skewed towards the common co-dependence of angles. The fast clash detection algorithm is based on K-d trees for intra- and inter-clashes of glycans. Standard grid mapping is used for the detection of clashes between glycans and their environment. To reduce the search space, the genetic algorithm iteratively optimizes subsets of glycans with the highest number of steric clashes.

Drawer class

Drawer class is used for generating 2D symbolic representations of glycans according to the IUPAC standard. The inter-residue connectivity graph stored in a Molecule is used for drawing the connectivity of a glycan. The protein is represented as a ribbon, each sequon is highlighted and the linked glycans are shown as a tree topology. The graphical representation is produced with Matplotlib and can be further modified by the users (e.g., add text, rescale) and exported in various image formats.

Results

Benchmark on viral glycoproteins

We compared the performance of Glycosylator and doGlycans, another Python framework for modeling glycans using three representative viral envelope glycoproteins, each containing different numbers of glycosylation sites and overall glycan density. The glycans on the surface of these proteins create a shield that helps them to evade the host's immune system [28]. For the benchmark, a mannose 9 was modeled at each sequon, mimicking the glycosylation state before exiting the endoplasmic reticulum [29]. The topology of the glycosylated structure was generated with the autopsf plug-in of VMD [30]. Each glycoprotein was then minimized with 5000 steps of conjugate gradient optimization in NAMD [31]. The resulting energy-minimized model was then submitted for a sanity check to pdb-care (http://www.glycosciences.de/tools/pdb-care/), a powerful tool that checks the connectivity and nomenclature in glycoproteins [32]. We observed that all glycoproteins modeled with Glycosylator had a lower potential energy and were devoid of any steric clashes and topological errors (Table 2). For structures with a low density of sequons, such as Influenza's hemagglutinin, Glycosylator and doGlycans performed similarly. However, a simple minimization was insufficient for removing steric clashes from the HIV-1 Envelope trimer and Delta coronavirus spike protein structures using doGlycans. The density of sequons at the surface of these glycoproteins is high, requiring a more effective strategy for removing clashes, such as provided by Glycosylator's Sampler Class. The steric clashes present in the structures produced with doGlycans lead topological errors, such as ring puckering after minimizations. In order to solve this issue, the torsional angles would have to be manually adjusted by the user.

Identifying and batch modeling *N*-linked glycans onto the HIV-1 Env trimer

As an additional test case, we modeled the glycan shield of the HIV-1 Env trimer using Glycosylator. The HIV-1 Env trimer consists of 80–100 sequons making it one of the most highly glycosylated proteins currently known. We chose the BG505-SOSIP structure with PDB: ID 5fyl, [33]) as the starting structure. First, all crystallographically-determined glycans were identified and hydrogenated (Fig. 1, upper left triangle). The ribbon representation allowed for a quick visual inspection of the identified *N*-linked sequons and linked glycans. A combination of mannose 5, mannose 9 and complex glycans was then modeled ab initio or by extending existing glycans to produce a more biologically relevant glycoform of the HIV-1 Env trimer (Fig. 1, lower right triangle). The Sampler function in Glycosylator was then used to remove all major clashes, such that the

Table 2 Benchmark comparing Glycosylator and doGlycans. The average minimum distance between sequons was computed between the closest pairs of asparagine Cα atoms. The number of issues accounts for errors in glycan connectivity and nomenclature due to steric clashes. The potential energy was calculated after 5000 steps of conjugate gradient energy minimization

Virus		Influenza A	HIV-1	Delta-coronavirus
PDB id		1 ha0	5fyl (gp120)	6bfu
Number of sequons		6	20	21
Average minimum distance between sequons	[Å]	21.60 ± 8.88	10.47 ± 4.74	14.71 ± 4.11
Glycosylator	Number of issues	0	**0**	**0**
	Potential energy [kcal/mol]	**− 9856.12**	**2524.05**	**− 5526.44**
DoGlycans	Number of issues	0	4	6
	Potential energy [kcal/mol]	− 9678.14	6055.20	− 1954.89

Better performance are highlighted in bold

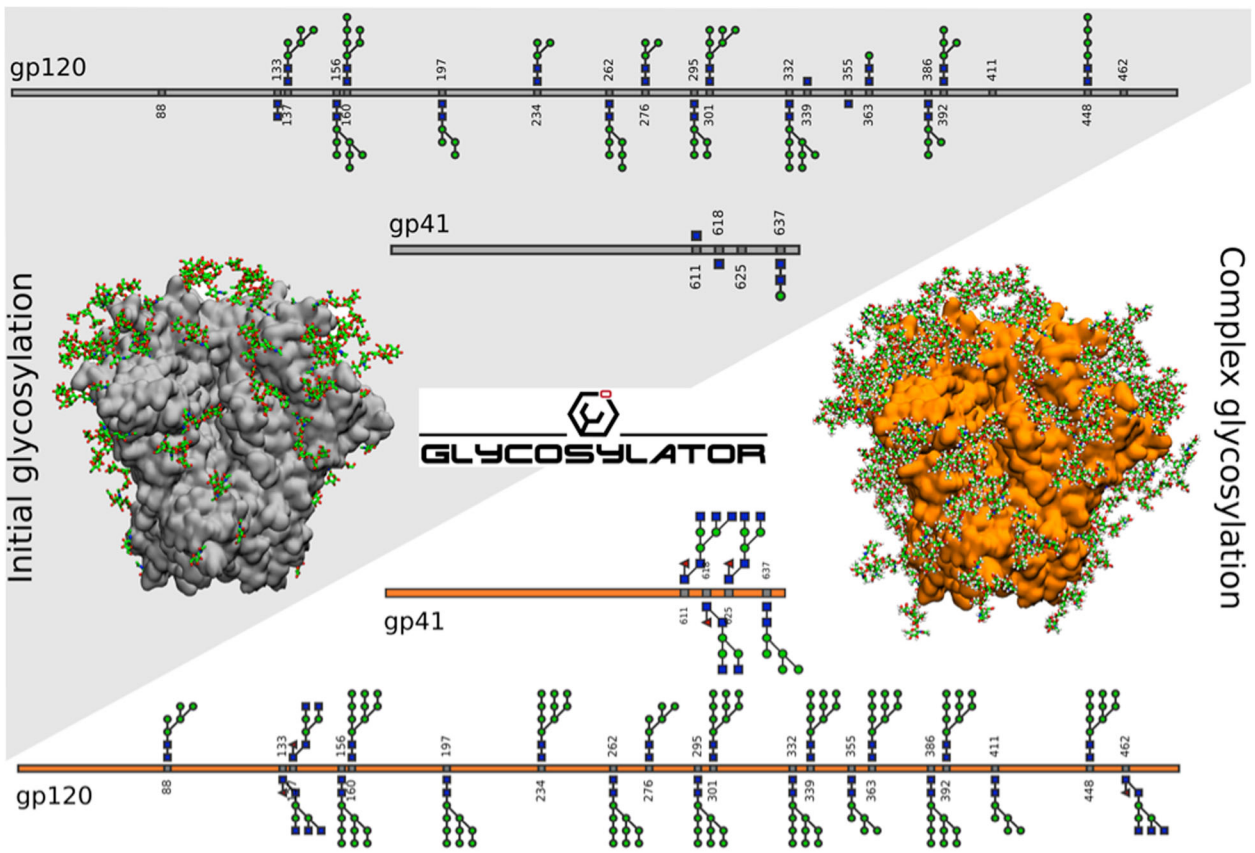


Fig. 1 Identification, visualization and modeling of N-linked glycans onto the HIV-1 Env trimer. Protein surface representation of the high-mannose glycoform of the HIV-1 Env Trimer (PDB ID: 5fyl). Crystallographically-determined glycans are shown in Licorice representation. Each subunit (gp120 and gp41) is represented as a ribbon with sequons indicated with gray squares and the N-linked glycans shown above or below the sequon (upper left triangular panel). Glycosylator was used to produce a complex glycoform variant of the HIV-1 Env trimer by modeling glycans ab initio or extending existing glycans (lower right triangular panel)

topology of the full glycoprotein could be generated directly with the autopsf plug-in of VMD [30]. The remaining clashes were quickly removed with 5000 steps of conjugate gradient energy minimization in NAMD [31]. The resulting model was then submitted to the pdb-care server [32] for a sanity check and we found no discrepancies in connectivity. The Python script used for this example is available in the GitHub repository. Two additional examples for building and identifying glycans can be found in the Supporting Information section (Additional file 1: Examples S1 and S2).

Conclusion

Glycosylator is a versatile Python framework for manipulating glycans and glycoproteins that facilitates the structural study of glycans. It will significantly improve the ability of the glycobiology community to model glycan structure without requiring advanced expertise in protein modeling or molecular dynamics. Glycosylator has already been successfully used for several studies investigating the dynamics of glycans over long timescales (500 ns to 2 μs) [33–35]. Glycosylator is a valuable asset for glycoinformatics and biomolecular modeling communities. Furthermore, it should be noted that Glycosylator can also be used to model other polymers (D09_polymer in Github).

Availability and requirements

Project name: Glycosylator.

Project home page: https://github.com/tlemmin/glycosylator

Operating system(s): Platform independent.

Programming language: Python.

License: MIT.

Supplementary information

Supplementary information accompanies this paper at https://doi.org/10.1186/s12859-019-3097-6.

Additional file 1: Figure S1. Architecture of Glycosylator, a Python framework for the rapid modeling of glycans. Each class is represented

by a hexagon. Full circles connecting classes indicate a class that contains an instance of the previous one as an attribute, e.g. instances of Molecule and MoleculeBuilder are attributes of Glycosylator. Several attributes from Glycosylator can be directly shared with Drawer and Sampler (white squares). Glycosylator can parse a PDB file of a glycoprotein and identify all the sequons (orange rhombus). The glycans (blue squares and green circles) will be extracted and saved as Molecule instances. Glycans at each sequon can then be built, modified or identified. **Figure S2.** Glycosylator Graphical User Interface. a) The main window is used to import a PDB file of a glycoprotein. Glycosylator will produce a symbolic representation (orange dashed line rectangle). A specific sequon can be selected in the right panel (purple dashed line rectangle). The glycan can be modified by clicking on the symbolic representation. b) The user can select a glycan from the common library or a library that they created. The selected glycan is highlighted with a red square. **Example S1.** Building a glycan. The structure of an N-Acetyl-D-Glucosamine will be imported as a Molecule instance. All missing atoms will be added according to the CHARMM force field. A second N-Acetyl-D-Glucosamine will then be linked through a 1-4 glycosidic bond. Finally, an Alpha-D-added according to the CHARMM force field. A second N-Acetyl-D-Glucosamine will then be linked through a 1-4 glycosidic bond. Finally, an Alpha-D-Mannosewill be built ab initio and saved to a PDB file. **Example S2.** Importing, identifying and editing of a glycan. The structure of a mannose 9 will be imported as a Molecule instance. A Glycosylator instance will then identify it against a database of known structures. Finally, the Molecule will be trimmed down to a mannose 6.

Abbreviations

API: Application Programming Interface; GUI: Graphical User Interface; NMR: Nuclear Magnetic Resonance; PDB: Protein Data Bank; UIPAC: International Union of Pure and Applied Chemistry

Acknowledgments

We would like to thank Dr. James E. Crowe for his support and comments on the manuscript, and Dr. Peter D. Kwong and Dr. Gwo-yu Chuang for their constructive feedback.

Authors' contributions

TL and CS conceived of the idea; TL designed and wrote the Glycosylator package; TL and CS wrote the paper. All authors read and approved the final manuscript.

Funding

This work was supported by a grant from the Human Vaccines Project. TL would like to acknowledge the support of the Swiss National Science Foundation (grant P3P3PA_174356). The funding bodies Human Vaccines Project and Swiss National Science Foundation had no role in the design of the study; collection, analysis, and interpretation of data; or in writing the manuscript.

Availability of data and materials

Glycosylator is available from the following Github repository: https://github.com/tlemmin/glycosylator

Ethics approval and consent to participate

Not applicable.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Author details

[1]DS3Lab, System Group, Department of Computer Sciences, ETH Zurich, CH-8093 Zurich, Switzerland. [2]Institute of Medical Virology, University of Zurich (UZH), CH-8057 Zurich, Switzerland. [3]Vanderbilt Vaccine Center, Vanderbilt University Medical Center, Nashville, TN 37232, USA. [4]Department of Pediatrics, Vanderbilt University Medical Center, Nashville, TN 37232, USA.

Received: 13 May 2019 Accepted: 12 September 2019
Published online: 22 October 2019

References

1. Apweiler R, Hermjakob H, Sharon N. On the frequency of protein glycosylation, as deduced from analysis of the SWISS-PROT database. Biochim Biophys Acta, Gen Subj. 1999;1473:4–8.
2. Yan A, Lennarz WJ. Unraveling the mechanism of protein N-glycosylation. J Biol Chem. 2005;280:3121–4.
3. Spiro RG. Protein glycosylation: nature, distribution, enzymatic formation, and disease implications of glycopeptides bonds. Glycobiology. 2002;12:43R–56R.
4. Gavel Y, von Heijne G. Sequence differences between glycosylated and non-glycosylated Asn-X-Thr/Ser acceptor sites: implications for protein engineering. Protein Eng Des Sel. 1990;3:433–42.
5. Xu C, Ng DTW. Glycosylation-directed quality control of protein folding. Nat Rev Mol Cell Biol. 2015;16:742–52.
6. Ferreira IG, Pucci M, Venturi G, Malagolini N, Chiricolo M, Dall'Olio F. Glycosylation as a main regulator of growth and death factor receptors signaling. Int J Mol Sci. 2018;19:580. https://doi.org/10.3390/ijms19020580.
7. Janik ME, Lityńska A, Vereecken P. Cell migration—the role of integrin glycosylation. Biochim Biophys Acta Gen Subj. 2010;1800:545–55.
8. Dell A, Galadari A, Sastre F, Hitchen P. Similarities and differences in the glycosylation mechanisms in prokaryotes and eukaryotes. Int J Microbiol. 2010. https://doi.org/10.1155/2010/148178.
9. Zhu F, Wu H. Insights into bacterial protein glycosylation in human microbiota. Sci China Life Sci. 2016;59:11–8.
10. Tate MD, Job ER, Deng Y-M, Gunalan V, Maurer-Stroh S, Reading PC. Playing hide and seek: how glycosylation of the influenza virus hemagglutinin can modulate the immune response to infection. Viruses. 2014;6:1294–316.
11. Vigerust DJ, Shepherd VL. Virus glycosylation: role in virulence and immune interactions. Trends Microbiol. 2007;15:211–8.
12. Tsuchiya S, Aoki NP, Shinmachi D, Matsubara M, Yamada I, Aoki-Kinoshita KF, et al. Implementation of GlycanBuilder to draw a wide variety of ambiguous glycans. Carbohydr Res. 2017;445:104–16.
13. Engelsen SB, Cros S, Mackie W, Pérez S. A molecular builder for carbohydrates: application to polysaccharides and complex carbohydrates. Biopolymers. 1996;39:417–33.
14. Danne R, Poojari C, Martinez-Seara H, Rissanen S, Lolicato F, Róg T, et al. doGlycans–tools for preparing carbohydrate structures for atomistic simulations of glycoproteins, glycolipids, and carbohydrate polymers for GROMACS. J Chem Inf Model. 2017;57:2401–6.
15. Bohne A, Lang E, von der Lieth CW. SWEET - WWW-based rapid 3D construction of oligo- and polysaccharides. Bioinformatics. 1999;15:767–8.
16. GLYCAM. http://glycam.org/. Accessed 5 Mar 2019.
17. Jo S, Song KC, Desaire H, MacKerell AD, Im W. Glycan reader: automated sugar identification and simulation preparation for carbohydrates and glycoproteins. J Comput Chem. 2011;32:3135–41.
18. Park S-J, Lee J, Patel DS, Ma H, Lee HS, Jo S, et al. Glycan reader is improved to recognize most sugar types and chemical modifications in the protein data Bank. Bioinformatics. 2017;33:3051–7.
19. Park S-J, Lee J, Qi Y, Kern NR, Lee HS, Jo S, et al. CHARMM-GUI glycan modeler for modeling and simulation of carbohydrates and glycoconjugates. Glycobiology. 2019;29:320–31.
20. Böhm M, Bohne-Lang A, Frank M, Loss A, Rojas-Macias MA, Lütteke T. Glycosciences.DB: an annotated data collection linking glycomics and proteomics data (2018 update). Nucleic Acids Res. 2019;47(Database issue): D1195–201.
21. Kirschner KN, Yongye AB, Tschampel SM, Gonzalez-Outeirino J, Daniels CR, Foley BL, et al. GLYCAM06: A generalizable biomolecular force field. Carbohydrates. J Comput Chem. 2008;29:622–55.
22. Guvench O, Mallajosyula SS, Raman EP, Hatcher E, Vanommeslaeghe K, Foster TJ, et al. CHARMM additive all-atom force field for carbohydrate derivatives and its utility in polysaccharide and carbohydrate-protein modeling. J Chem Theory Comput. 2011;7:3162–80.
23. Jo S, Kim T, Iyer VG, Im W. CHARMM-GUI: a web-based graphical user interface for CHARMM. J Comput Chem. 2008;29:1859–65.
24. Lee J, Patel DS, Ståhle J, Park S-J, Kern NR, Kim S, et al. CHARMM-GUI membrane builder for complex biological membrane simulations with glycolipids and lipoglycans. J Chem Theory Comput. 2019;15:775–86.
25. Bakan A, Meireles LM, Bahar I. Prody: protein dynamics inferred from theory and experiments. Bioinformatics. 2011;27:1575–7.

26. Hagberg AA, Schult DA, Swart PJ. Exploring network structure, dynamics, and function using NetworkX, in Proceedings of the 7th Python in Science Conference (SciPy2008), Gäel Varoquaux, Travis Vaught, and Jarrod Millman (Eds), Pasadena; 2008. p. 11–5.
27. Hunter JD. Matplotlib: a 2D graphics environment. Comput Sci Eng. 2007;9:90–5.
28. Le Mercier P, Mariethoz J, Lascano-Maillard J, Bonnardel F, Imberty A, Ricard-Blum S, et al. A bioinformatics view of glycan–virus interactions. Viruses. 2019;11:374.
29. Schierbaum F. Comprehensive Glycoscience (From Chemistry to Systems Biology). By Johannis P. Kamerling (Editor-in-Chief), Geert-J. Boons, Yuan Ch. Lee, Akemi Suzuki, Naoyuki Taniguchi, Alphons G.J. Voragen. Starch - Stärke. 2008;60:48–9.
30. Humphrey W, Dalke A, Schulten K. VMD: visual molecular dynamics. J Mol Graph. 1996;14:33–8.
31. Phillips JC, Braun R, Wang W, Gumbart J, Tajkhorshid E, Villa E, et al. Scalable molecular dynamics with NAMD. J Comput Chem. 2005;26:1781–802.
32. Lütteke T, von der Lieth C-W. pdb-care (PDB CArbohydrate REsidue check): A program to support annotation of complex carbohydrate structures in PDB files. BMC Bioinformatics. 2004;5:69.
33. Stewart-Jones GBE, Soto C, Lemmin T, Chuang G-Y, Druz A, Kong R, et al. Trimeric HIV-1-Env structures define glycan shields from clades a, B, and G. Cell. 2016;165:813–26.
34. Shahzad-Ul-Hussan S, Sastry M, Lemmin T, Soto C, Loesgen S, Scott DA, et al. Insights from NMR spectroscopy into the conformational properties of Man-9 and its recognition by two HIV binding proteins. Chembiochem. 2017;18:764–71.
35. Lemmin T, Soto C, Stuckey J, Kwong PD. Microsecond dynamics and network analysis of the HIV-1 SOSIP Env trimer reveal collective behavior and conserved microdomains of the glycan shield. Structure. 2017;25:1631–1639.e2.

Publisher's Note